(19) JAPANESE PATENT OFFICE

PATENT ABSTRACTS OF JAPAN

(11) Publication number: 2000057199 A

(43) Date of publication of application: 25 . 02 . 00

(51) Int. Cl G06F 17/50
H01L 21/82
H01L 27/04
H01L 21/822

(21) Application number: 10229840

(22) Date of filing: 14 . 08 . 98

(71) Applicant: TOSHIBA CORP

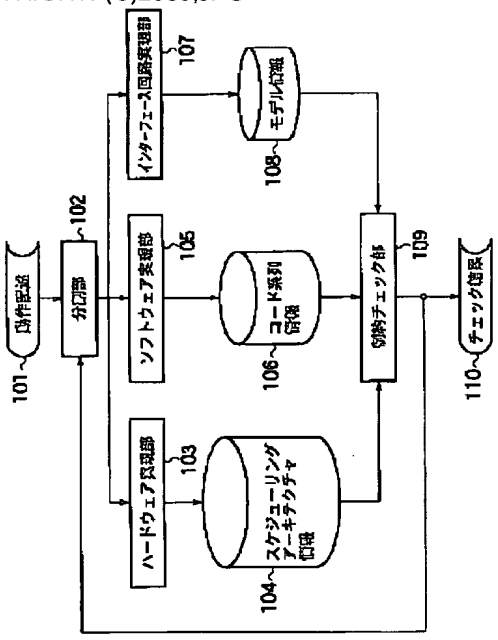
(72) Inventor: IMAI HIROSHI

(54) DEVICE FOR SUPPORTING DESIGN OF SYSTEM AND COMPUTER- READABLE RECORDING MEDIUM RECORDING DESIGN SUPPORT PROGRAM

(57) Abstract:

PROBLEM TO BE SOLVED: To make easily and quickly designable more correct system by considering influence of an interface means for transferring data between hardware and software at the time of designing a system by sharing processing functions between the hardware and the software to execute evaluation.

SOLUTION: The system design supporting device shares an operation description 101 describing the processing functions of a system between a hardware realizing part 103 and a software realizing part 105, evaluates the processings sharedly by the hardware and the software, checks design restriction by a restriction checking part 109 according to the result of the evaluation and designs the system while adjusting the share by a dividing part 102. The evaluation concerning the processing of an interface means for transferring the processing result between the hardware and the software is executed by an interface circuit realizing part 107 at the time of adjustment and the result is reflected.



(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号
特開2000－57199
（P2000－57199A）
(43)公開日　平成12年２月25日(2000.2.25)

(51)Int.Cl.⁷	識別記号	FI		テーマコード（参考）
G06F 17/50		G06F 15/60	654R	5B046
H01L 21/82			636G	5F038
27/04			654D	5F064
21/822		H01L 21/82	C	
		27/04	U	

審査請求　未請求　請求項の数２　ＯＬ　（全 15 頁）

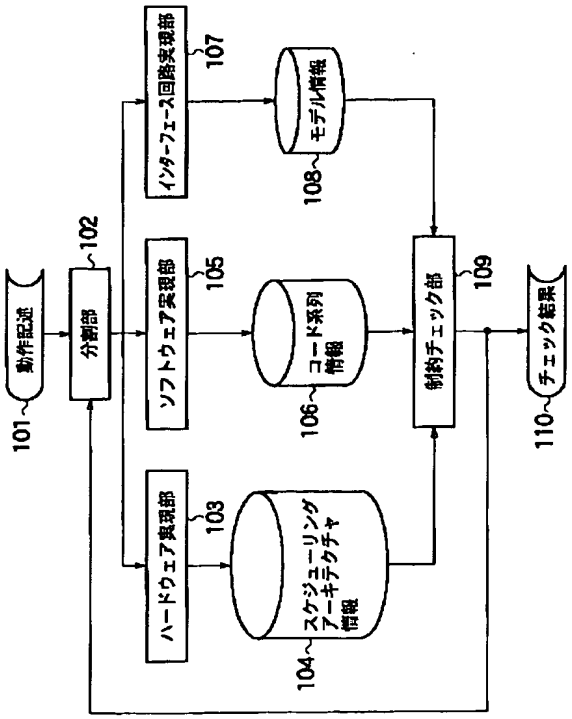
(21)出願番号　特願平10－229840
(22)出願日　平成10年８月14日(1998.8.14)

(71)出願人　000003078
株式会社東芝
神奈川県川崎市幸区堀川町72番地
(72)発明者　今井　浩史
神奈川県川崎市幸区堀川町580番１号　株式会社東芝半導体システム技術センター内
(74)代理人　100083806
弁理士　三好　秀和　（外３名）

Ｆターム(参考)　5B046 AA08 BA02
5F038 CA17 DF14 EZ09 EZ10 EZ20
5F064 AA01 BB01 DD02 DD03 EE02
EE03 EE57 EE58 HH06 HH09
HH11 HH12

(54)【発明の名称】　システムの設計支援装置及び設計支援プログラムを記録したコンピュータ読み取り可能な記録媒体

(57)【要約】
【課題】　処理機能をハードウェアとソフトウェアに分担して評価を行うことによってシステムの設計を行う際に、ハードウェアとソフトウェア間でデータの受け渡しを行うインターフェース手段による影響を考慮することによって、より正確なシステムの設計を容易且つ迅速に行う。
【解決手段】　本発明のシステム設計支援装置は、システムの処理機能を記述した動作記述１０１を、ハードウェア実現部１０３とソフトウェア実現部１０５とに分担し、この分担に基づいてハードウェアとソフトウェアによる処理を評価し、この評価の結果によって、制約チェック部１０９で設計制約をチェックし、これにより分割部１０２で分担を調整しつつシステムの設計を行うものであって、調整の際に、ハードウェアとソフトウェアとの間における処理結果の受け渡しを行うインターフェース手段の処理に関する評価をインターフェース回路実現部１０７で行い、この結果を反映させるものである。



1

【特許請求の範囲】

【請求項１】　システムの処理機能をハードウェアとソフトウェアに分担し、この分担に基づいて該ハードウェアと該ソフトウェアによる処理を評価し、この評価の結果によって前記分担を調整しつつ前記システムの設計を行うシステム設計支援装置であって、

前記調整の際に、前記ハードウェアと前記ソフトウェアとの間における処理結果の受け渡しを行うインターフェース手段の処理に関する評価結果を反映させることを特徴とするシステムの設計支援装置。

【請求項２】　システムの処理機能をハードウェアとソフトウェアに分担し、この分担に基づいて該ハードウェアと該ソフトウェアによる処理を評価し、この評価の結果によって前記システムの設計を行うシステム設計支援プログラムを記録したコンピュータ読み取り可能な記録媒体であって、

システムの処理機能を記述した動作仕様を入力するステップと、

前記動作仕様に基づいて前記処理機能をハードウェアとソフトウェアに分担するステップと、

前記ハードウェアに分担された動作仕様について評価を行うステップと、

前記ソフトウェアに分担された動作仕様について評価を行うステップと、

前記ハードウェアと前記ソフトウェアとの間における処理結果の受け渡しを行うインターフェース手段の処理に関する評価を行うステップと、

前記ハードウェア、ソフトウェア及びインターフェースについての各評価結果に基づいて、前記システムの設計制約を満たすか否かの判断を行うステップと、

前記判断において前記設計制約を満たさない場合には、前記各評価結果に基づいて前記分担を調整するステップとを有することを特徴とするシステム設計支援を記録したコンピュータ読み取り可能な記録媒体。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、コンピュータ等の演算処理システムのハードウェア・ソフトウェア・コデザインに関し、処理動作をハードウェア及びソフトウェアに分割したうえで、これらの処理を評価し、その評価結果が設計制約を満足するかどうかチェックすることによってシステムの設計を支援する装置及びそのプログラムを記録したコンピュータ読み取り可能な記録媒体に関するものである。

【０００２】

【従来の技術】近年、ＡＳＩＣ、ＣＰＵ、メモリ等から構成される複合演算処理システム開発の要求が高まっている。このようなシステムを設計するには、システム全体の処理動作を記述した仕様をＡＳＩＣ等にハードウェア化する部分、ＣＰＵで実行されるソフトウェアにする

2

部分へ分割する必要がある。これをハードウェア・ソフトウェア分割という。

【０００３】かかるハードウェア・ソフトウェア分割の際には、ＡＳＩＣ、ＣＰＵ、メモリ等のシステム全体の面積、性能等のコストを見積もり、設計者が与えた回路規模や性能等の設計制約を満たすように動作仕様をハードウェア・ソフトウェアに分担する必要がある。

【０００４】通常、一回の分割処理で制約を満たすハードウェア・ソフトウェア分割が見つかることは希である。従って、いくつかの分割を試し、設計制約を満たすハードウェア・ソフトウェア分割を見つける。設計制約を満たすハードウェア・ソフトウェア分割を見つけるためには、システム全体の面積、性能等のコストをいかに正確に見積もるかが重要となる。なぜなら、正確に見積もらなかった場合、見積もりの段階では設計制約を満たしていたにもかかわらず、設計段階が進むにつれ実際の値と見積もり値との差が大きくなり、設計制約を満たさないことが分かったときには、分割のやり直しが必要となり、設計期間が長期化する惧れがある。

【０００５】さらに、ハードウェア・ソフトウェア分割は複数回行われるので、設計期間を短縮するためには面積、性能等のコストの見積もり時間を短くする必要がある。

【０００６】

【発明が解決しようとする課題】しかし、従来、例えばタイミング制約を満足するかどうかチェックする際に、ハードウェアで実現される部分のスケジューリング結果とソフトウェアで実現される部分のソフトウェア規模にのみ基づいてチェックしているため正確なチェックができなかった。従来の見積もり方法については、例えば、R.K.Gupta著、Co-Synthesis of Hardware and Software for Digital Embedded Systems, KluwerAcademic Publishers, 1995に説明されている。

【０００７】また、正確なチェックを行うためには、ＡＳＩＣ等のハードウェアとＣＰＵ間のデータの受け渡しに必要なインターフェース回路を設計し、シミュレーションを行う必要があるため、チェックに時間が掛かる。

【０００８】一方、設計期間を短縮するため正確さを犠牲にしたチェック方法を採用した場合、設計段階が進み、選択されたハードウェア・ソフトウェア分割に必要なインターフェース回路が設計される段階になると、面積、性能の値と見積もりとの差が大きくなり、意図した回路とは程遠い回路しか設計できないといった問題があった。

【０００９】さらには、正確な設計制約を行うため、設計制約チェックにシミュレーションを用いていたのでは、設計制約チェックに時間が掛かり、設計期間が長期化してしまうといった問題があった。

【００１０】そこで、本発明は上記事情に鑑みて成されたものであり、その目的は、処理機能をハードウェアと

ソフトウェアに分担して評価を行うことによってシステムの設計を行う際に、ハードウェアとソフトウェア間でデータの受け渡しを行うインターフェース手段による影響を考慮することによって、より正確なシステムの設計を容易且つ迅速に行うことができるシステムの設計支援装置及び設計支援プログラムを記録したコンピュータ読み取り可能な記録媒体を提案することにある。

【００１１】

【課題を解決するための手段】上記課題を解決するために請求項１に記載の発明は、システムの処理機能をハードウェアとソフトウェアに分担し、この分担に基づいて該ハードウェアと該ソフトウェアによる処理を評価し、この評価の結果によって前記分担を調整しつつ前記システムの設計を行うシステム設計支援装置であって、前記調整の際に、前記ハードウェアと前記ソフトウェアとの間における処理結果の受け渡しを行うインターフェース手段の処理に関する評価結果を反映させるものである。

【００１２】このような請求項１に記載の発明によれば、ハードウェアとソフトウェアとを分割する際に、これらの間で処理結果の受け渡しを行うインターフェース手段の影響を考慮するため、より正確なシステムの設計を行うことができる。

【００１３】また、請求項２に記載の発明は、システムの処理機能をハードウェアとソフトウェアに分担し、この分担に基づいて該ハードウェアと該ソフトウェアによる処理を評価し、この評価の結果によって前記システムの設計を行うシステム設計支援プログラムを記録したコンピュータ読み取り可能な記録媒体であって、システムの処理機能を記述した動作仕様を入力するステップと、前記動作仕様に基づいて前記処理機能をハードウェアとソフトウェアに分担するステップと、前記ハードウェアに分担された動作仕様について評価を行うステップと、前記ソフトウェアウェアに分担された動作仕様について評価を行うステップと、前記ハードウェアと前記ソフトウェアとの間における処理結果の受け渡しを行うインターフェース手段の処理に関する評価を行うステップと、前記ハードウェア、ソフトウェア及びインターフェースについての各評価結果に基づいて、前記システムの設計制約を満たすか否かの判断を行うステップと、前記判断において前記設計制約を満たさない場合には、前記各評価結果に基づいて前記分担を調整するステップとを有するものである。

【００１４】このような請求項２に記載の発明によれば、システムの設計を正確にもしくは迅速に行えるという有用な設計支援プログラムの保存、実行、運搬等が容易に行うことができ、より簡便にシステムの設計を行うことができる。

【００１５】

【発明の実施の形態】以下、この発明に係るシステムの設計支援装置の実施形態について説明する。

【００１６】（１）実施例１

本発明の第１の実施形態を図面を参照して説明する。図１は本実施形態に係るハードウェア・コデザイン装置の構成図である。同図において、設計対象の動作仕様を記述した動作記述１０１が、分割部１０２においてＡＳＩＣ等のハードウェアで実現される部分と、ＣＰＵ上で実行されるソフトウェアで実現される部分とに分割される。この分割は面積と性能の設計制約を満たす分割が得られるまで様々な分割を試し、繰り返される。

【００１７】ハードウェア実現部１０３では、ハードウェアに分担された動作記述部分のハードウェア実現情報であるスケジューリング・アーキテクチャ情報１０４が生成される。このスケジューリング情報とは、例えばハードウェアで実現される動作記述部分の実行を完了するまでの時間の情報等であり、また、アーキテクチャ情報とは、ハードウェアの構造、例えば、パイプライン構造等の情報等である。これらの情報からハードウェアへのデータの受け渡しの情報が得られる。

【００１８】一方、ソフトウェア実現部１０５では、ＣＰＵ上で実行されるソフトウェアとして実現される動作記述部分のコードの規模、実行時間を表すコード系列情報１０６が生成される。

【００１９】他方、インターフェース回路実現部１０７では、ＡＳＩＣ等で実現されるハードウェアとソフトウェアを実行するＣＰＵ間のデータの受け渡しに必要なインターフェース回路を生成する。さらに、このインターフェース回路実現部１０７は、インターフェース回路の動作の特徴を表すモデル情報１０８を生成する。

【００２０】このモデル情報１０８は、例えば、ハンドシェイク等のインターフェース・プロトコルのタイプ、データ受け渡しから結果を得るまでの時間、次のデータを受け渡せるまでの待ち時間等である。

【００２１】そして、制約チェック部１０９において、ハードウェアの実現情報であるスケジューリング・アーキテクチャ情報１０４と、ＣＰＵ上で実行されるソフトウェアの規模と実行時間の情報であるコード系列情報１０６と、ハードウェアとＣＰＵ間のインターフェース回路の動作の情報であるモデル情報１０８とに基づいて、ハードウェア・ソフトウェア分割が実行時間等の設計制約を満足するかどうかがチェックされ、チェック結果１１０が出力される。

【００２２】このチェック結果１１０は分割部１０２へフィードバックされ、ハードウェア・ソフトウェア分割の再試行に利用される。

【００２３】こののような操作を行うことにより、ハードウェア・ソフトウェア分割の繰り返しの際に、その分割が設計制約を満足するかどうか正確かつ短時間にチェックすることが可能となり、正確な見積もりに基づいたハードウェア・ソフトウェア分割が短時間に得られる。

【００２４】（２）実施形態２

次に本発明の第２の実施形態について説明する。この実施例は第１の実施例で示したハードウェア実現部におけるスケジューリング・アーキテクチャ情報生成を実現するものである。図２にスケジューリング・アーキテクチャ情報生成のフローチャートを示す。

【００２５】同図において、ステップ２０１でＡＳＩＣ等のハードウェアで実現する部分の動作記述を読み込む。次いで、ステップ２０２でハードウェアのアーキテクチャを選択する。この選択とは、例えば、パイプライン構造を採用する、採用しない等である。

【００２６】ステップ２０３でステップ２０２で選択されたアーキテクチャに関する情報が生成される。その後、ステップ２０４で動作記述をハードウェアで実現するために必要な回路部品の種類と個数を決定するアロケーションと呼ばれる処理を実行する。次いで、ステップ２０５でステップ２０４で決定した回路部品の種類と個数に基づいて、動作記述をクロック・サイクルごとに実行される動作系列に分割するスケジューリングと呼ばれる処理を実行する。

【００２７】ステップ２０６では、ステップ２０５で実行されたスケジューリング結果に基づいてスケジューリング情報を生成する。ステップ２０７でステップ２０５でスケジューリング処理されたクロック・サイクル毎の各演算を実際の回路部品に割り当てるバインディングと呼ばれる処理を実行する。

【００２８】ステップ２０８でハードウェア記述を生成する。このハードウェア記述に基づいてハードウェアが合成される。動作記述からハードウェア記述を生成する方法は、例えば、D.Gajski他著、High-Level Synthesis:Introduction to Chip andSystem Design, Kluwer Academic Publishers, 1992が詳しい。

【００２９】図３に動作記述の例を示す。また、ハードウェア・ソフトウェア分割の例を図４に示す。この例では、動作記述中の関数ｆ（ｘ）がハードウェアで実現され、関数main()がソフトウェアで実現される。また、図３に示した動作記述中の関数ｆ（ｘ）の動作を表す動作記述部分がステップ２０１で読み込まれる。

【００３０】この動作記述に対してステップ２０５で実現されるスケジューリング結果の例を図５に示す。この例では、関数ｆ（ｘ）の実行を完了するのに１０クロック・サイクル掛かることが分かる。また、図６に前述したステップ２０３とステップ２０６で生成されるスケジューリング・アーキテクチャ情報の例を示す。アーキテクチャ項目は、ハードウェアがパイプライン構造をとるかどうかを表している。

【００３１】ここで、パイプライン・ピッチ項目は、各ステージの実行時間（Ｃ）を表す。ステージ数項目は、パイプラインの段数（Ｎ）を表す。レイテンシー項目は、１組のデータが入力されてから結果が得られるまでのクロック・サイクル数（Ｌ）を表す。データ投入サイクル項目は、ハードウェアにデータを受け渡せる間隔を表す。この値は、ステージ数とレイテンシーの値から、Ｌ／Ｎで計算できる。

【００３２】例えば、図３に示した動作記述に対して、図４に示したハードウェア・ソフトウェア分割が行われた場合で、アーキテクチャとして非パイプラインを選択し、図５に示したスケジューリング結果の場合のスケジューリング・アーキテクチャ情報の例を図７に示す。本実施形態では、アーキテクチャ項目は、非パイプライン構造を選択したので、非パイプラインとなっている。ステージ数項目は、非パイプラインなので１となる。レイテンシー項目は、スケジューリング結果が１０クロック・サイクルとなっているので、１０となる。

【００３３】データ投入サイクル項目は、非パイプラインでハードウェアが構成されるので、１組のデータの処理が終わるまで次のデータをハードウェアに受け渡すことができず、１０となる。また、アーキテクチャとして、ステージ数１０のパイプラインを選択した場合のスケジューリング・アーキテクチャ情報の例を図８に示す。

【００３４】本実施形態では、かかるアーキテクチャ項目は、パイプライン構造を選択したので、パイプラインとなる。また、ステージ数項目は、１０ステージのパイプラインを選択したので、１０となる。レイテンシー項目は、スケジューリング結果が１０クロック・サイクルとなっているので、１０となる。データ投入サイクル項目は、１０ステージのパイプラインを選択したので、１となる。

【００３５】上記のようにスケジューリング・アーキテクチャ情報が作成される。

【００３６】（３）実施形態３

次に本発明の第３の実施形態について説明する。この実施例は第１の実施例で示したソフトウェア実現部におけるコード系列情報生成を実現するものである。図９にコード情報系列情報生成のフローチャートを示す。

【００３７】同図において、ステップ９０１で、ハードウェア・ソフトウェア・コデザインの動作記述中ＣＰＵ上で実行されるソフトウェアで実現される動作記述部分が読み込まれる。ステップ９０２で、ステップ９０１で読み込んだ動作記述を基本命令に分割する。ステップ９０３で、ステップ９０２で基本命令に分割された動作記述に基づいてＣＰＵ上で実行される基本命令のコード系列を生成する。ステップ９０４で、ステップ９０３で生成されたコード系列に基づいてコード系列情報が生成される。ステップ９０５で、ＣＰＵ上で実行されるソフトウェアが生成される。動作記述からソフトウェアを生成する方法は、R.Gupta著、Co-Synthesisof Hardware and Software for Digital Embedded Systems, Kluwer Academic Publishers, 1995が詳しい。

【００３８】前述した図３に示した動作記述の例に対し

て図４に示したハードウェア・ソフトウェア分割が行われた場合のソフトウェアで実現される関数main()のコード系列の例を図１０に示す。図１１にステップ９０４で生成されるコード系列情報の例を示す。コード系列情報として、ＣＰＵ上でこのソフトウェアを実行するために必要な時間を表すステップ数とハードウェアで実現された関数の呼び出し回数が生成される。

【００３９】また、図１０に示したコード系列の例に対するコード系列情報の例を図１２に示す。図１０の例では、ハードウェアで実現される関数ｆ（ｘ）の呼び出しが２回あるので、関数呼び出し回数欄が２となっている。

【００４０】（４）実施形態４

次に本発明の第４の実施形態について説明する。この実施形態では第１の実施例で示したインターフェース回路実現部におけるモデル情報生成を実現するものである。図１３にＡＳＩＣ等のハードウェアとソフトウェアを実行するＣＰＵ間に必要なインターフェース回路のモデル情報生成のフローチャートを示す。

【００４１】同図において、ステップ１３０１でインターフェース回路情報を読み込む。ステップ１３０２でステップ１３０１で読み込んだインターフェース回路情報からインターフェース回路のモデル情報を生成する。ステップ１３０３でインターフェース回路の動作を表す記述が生成される。この記述が合成され、インターフェース回路が実現される。

【００４２】図１４にモデル情報の例を示す。この情報は、インターフェース回路のプロトコル・タイプ、データを渡してから結果が得られるまでの時間、データ投入間隔を表すデータ投入サイクルからなる。プロトコル・タイプとしては、例えば、ハンドシェイク、キュー等がある。データを渡してから結果が得られるまでの時間は、ハードウェアの実行時間であり、アーキテクチャ・スケジューリング情報のレイテンシー項目のＬとなる。データ投入サイクルは、ハンドシェイク・タイプのインターフェースの場合、１度結果が得られるまで次のデータを渡せないので、、Ｌとなる。

【００４３】キュー・タイプのインターフェースの場合、データをインターフェース回路で保持しておくことができるため、結果が得られるまで待つ必要はなく、１となる。図３に示した動作記述に対して図４に示したハードウェア分割が行われた場合のモデル情報の例を図１５に示す。ハードウェアがパイプラインで実現されているかいないかに関わらず、結果が得られるまでの時間は１０、データ投入サイクルは、ハンドシェイク・タイプの場合は１０、キュー・タイプの場合は１となる。

【００４４】上記のようにインターフェースのモデル情報が生成される。

【００４５】（５）実施形態５

次に本発明の実施形態について説明する。この実施形態では第１の実施例で示した制約チェック部におけるハードウェア・ソフトウェア分割が設計制約を満たすかどうかの判断を実現するものである。図１６に制約チェックのフローチャートを示す。

【００４６】同図において、ステップ１６０１でハードウェア・ソフトウェア・コデザインの設計制約を読み込む。ステップ１６０２でハードウェア実現部１０２が生成したスケジューリング・アーキテクチャ情報１０３を読み込む。ステップ１６０３でソフトウェア実現部１０４が生成したコード尾系列情報１０５を読み込む。ステップ１６０４でインターフェース回路実現部１０６が生成したモデル情報１０７を読み込む。ステップ１６０５でステップ１６０２から１６０４で読み込んだ情報に基づいて制約の評価値を計算する。

【００４７】ステップ１６０６でステップ１６０５で計算した評価値とステップ１６０１で読み込んだ設計制約を比較する。評価値が設計制約を満たす場合、ステップ１６０７で設計制約を満たしたことを表示する。評価値が設計制約を満たさない場合、ステップ１６０８で設計制約を満たさないことを表示する。

【００４８】前述した図３に示した動作記述に対して図４に示したハードウェア・ソフトウェア分割が行われ、インターフェース回路としてキュー・タイプのインターフェース回路を仮定した場合のタイミング制約評価値の計算について説明する。

【００４９】ここで、ソフトウェアを実行するＣＰＵのクロック・サイクルをC_1、関数ｆ（ｘ）を実現するハードウェアのクロック・サイクルをC_2とする。また、$C_1 < C_2$と仮定する。関数main()中で、ｘとｙの値はハードウェアが計算結果を返すのに必要な時間、すなわち、$C_2 \times L$より十分後に参照されると仮定する。

【００５０】まず、ソフトウェアの実行にかかる時間は、図１２に示したコード系列情報から選れるステップ数ｎとＣＰＵのクロック・サイクルC_1との積$C_1 \times n$で与えられる。次に、関数ｆ（ｘ）の呼び出しに必要な時間を計算する。図１２に示したコード系列情報から、関数ｆ（ｘ）の呼び出しは２回であることが分かる。さらに、２回のｆ（ｘ）の呼び出しは連続しており、その呼び出し間に待ち時間が必要である。

【００５１】この時間を図１５に示したインターフェース回路のモデル情報から計算する。インターフェースのプロトコルがハンドシェイク・タイプの場合、モデル情報のデータ投入サイクル項目の値１０とハードウェアのクロック・サイクルC_2とから、この時間は$C_2 \times 10$と計算される。インターフェースのプロトコルがキュー・タイプの場合、モデル情報のデータ投入サイクル項目の値１とハードウェアのクロック・サイクルC_2とから、この時間は$C_2 \times 1$と計算される。

【００５２】本実施形態では、インターフェースのプロトコルとしてキュー・タイプを仮定しており、また、Ｃ

$_1 < C_2$なので、２回の関数ｆ（ｘ）の呼び出し間の時間を考慮する必要はない。したがって、タイミング制約の評価値は図１７に示した値となる。なお、従来のように、インターフェース回路のモデルを考慮しない場合、２回の関数ｆ（ｘ）の呼び出し間の時間を$C_2 \times L$と評価してしまい、タイミング制約の評価値は図１８に示すような結果となってしまい、正確なチェックができない。上記のようにハードウェア・ソフトウェア分割が設計制約を満たすかどうかチェックされる。

【００５３】さらに、本発明に係るシステムの設計支援装置の他の実施形態について説明する。

【００５４】（６）実施形態６

本発明の第６の実施形態を図面を参照して説明する。図１９に本実施形態に係るハードウェア／ソフトウェア・コデザイン装置の構成図を示す。

【００５５】同図において、ハードウェア／ソフトウェア分割部１９０１で、入力された動作仕様がＡＳＩＣ等のハードウェアで実現される部分と、ＣＰＵで実行されるソフトウェアで実現される部分とに分割される。この分割は面積と性能の制約を満たす分割が得られるまで繰り返される。

【００５６】また、インターフェース回路詳細化部１９０２では、ハードウェアとソフトウェアを実行するＣＰＵ間等のデータの受渡しのために必要なインターフェース回路を詳細化する。この詳細化の処理は分割探索の期間でただ一度だけ実行される。

【００５７】差分情報生成部１９０３では、分割探索期間を通じて、インターフェース回路を詳細化した分割と現分割との差分情報を生成する。この差分情報は現分割に必要となるインターフェース回路の面積と性能を見積もる際に利用される。

【００５８】評価部１９０４では、前記インターフェース回路詳細化部１９０２で詳細化されたインターフェース回路と、前記差分情報生成部１９０３で生成された差分情報に基づいて、現分割の面積と性能の見積もりを実行する。この見積もりには、現分割に必要なインターフェース回路の面積と性能の見積もりをも含む。

【００５９】前記ハードウェア／ソフトウェア分割部１９０１は、前記評価部１９０４が生成した見積もりに従って、最終的に最適な分割を選択し、ハードウェア仕様、ソフトウェア仕様、インターフェース仕様を出力する。

【００６０】このような操作を行うことにより、最適な分割の探索期間の各分割に対して面積と性能を見積もる際に、正確な見積もりが短期間に可能となり、正確な見積もりに基づいたハードウェア／ソフトウェア分割が得られる。

【００６１】（７）実施形態７

次に本発明の第７の実施形態例について説明する。この実施形態では第６の実施形態で示したインターフェース回路詳細化部におけるインターフェース回路の詳細化を実現するものである。

【００６２】動作仕様の例を図２０に示す。この動作仕様に対しハードウェア／ソフトウェア分割部１９０１に於いて、１つのハードウェア／ソフトウェア分割が生成される。そのアルゴリズムは、例えば、R.K.Gupta, “Co-Synthesis of Hardware andSoftware for Digital Embedded Systems,” Kluwer Academic Publishers, 1995. に見つけられる。

【００６３】ハードウェア／ソフトウェア分割の例を図２１に示す。この例では、図２０に示した動作仕様中、関数ｆ（ａ，ｂ）で表された部分がハードウェアで実現され、動作仕様の残りの部分がＣＰＵ上で実行されるソフトウェアで実現される。

【００６４】ハードウェアで実現される動作仕様部分は、例えば、高位合成アルゴリズムによって実現される。高位合成アルゴリズムについては、D.D.Gajski, et al, “High-Level Synthesis:Introduction to Chip and System Design,” Kluwer Academic Publishers, 1992. に見つけられる。

【００６５】図２１に示したハードウェア／ソフトウェア分割候補の場合、ハードウェアで実現されるｆ（ａ，ｂ）のハードウェアの動作を図２２に示す。図２２では、ハードウェアの動作をデータ・フロー・グラフ（ＤＦＧ）で表しており、横線で区切られ、番号付けられた各ステップはステートと呼ばれ、横線はクロック・サイクルの境界を表す。

【００６６】同図において、各ステートはクロックに従って順に実行される。ＤＦＧ中マルは演算を表し、黒く塗りつぶされた矩形はレジスタを表す。ｆ（ａ，ｂ）を実現したハードウェアは、３クロック・サイクル掛かって動作を完了する。さらに、ＣＰＵがハードウェアへデータを受け渡すタイミング、ＣＰＵがハードウェアからデータを受け取るタイミングを示した。

【００６７】図２２に示した例では、変数ａに対応したデータがＣＰＵからステートに受け渡され、ハードウェアが変数ａにステート１でデータを読み込んでいる。なお、変数ｂについても同様である。変数ｏｕｔの値がハードウェアからステート３で出力され、変数ｏｕｔに対応したデータをＣＰＵはステート３で読み込んでいる。

【００６８】また、インターフェース回路詳細化部１９０２では、インターフェース回路詳細化が実行され、インターフェース回路詳細化情報テーブルが生成される。このインターフェース回路詳細化情報テーブル作成のフローチャートを図２３に示す。

【００６９】初めに、ステップ２３０１と２３０２で、ハードウェアで実現される動作仕様の入力変数と出力変数のリストを作成する。次に、ステップ２３０３で入力変数リストＬＩから変数を１つ取り出す。その変数をｖとする。ステップ２３０４でその変数ｖを取り付けるポ

ート番号を選択する。

【００７０】この選択は、自動的に行っても、設計者が自ら手動で行ってもよい。選択されたポート番号がインターフェース回路詳細化情報テーブルに登録されていない場合、まず、ステップ２３０５でそのポート番号をテーブルに登録する。さらに、ステップ２３０６から２３１３で、種類欄、ビット幅欄、ステート欄、バッファ欄、変数欄に、それぞれ、“入力”、変数ｖのビット幅、変数ｖが読み込まれるステート番号、バッファの有無、変数名ｖを登録する。

【００７１】選択されたポート番号が既にテーブルに登録されている場合、変数ｖのビット幅とテーブルのビット幅欄の値を比較して、変数ｖのビット幅の方が大きい場合、ステップ２３１５で変数ｖのビット幅でビット幅欄を更新する。そして、変数ｖが読み込まれるステートと変数ｖを、それぞれ、ステップ２３１４と２３１６でテーブルのステート欄と変数欄に追加する。なお、出力変数リストについても同様な処理を行う。図２２に示した例に対するインターフェース回路詳細化情報テーブルを図２５に示す。

【００７２】次に、前述したステップ２３１０で実行されるバッファの有無の判断の処理について説明する。この判断のフローチャートを図２４に示す。まず、ステップ２４０１で対象となっている変数が入力変数か出力変数か調べる。対象となる変数が入力変数であった場合、ステップ２４０２でＣＰＵから変数に対応したデータが与えられるステート（ＳＴｉ）を調べる。次に、ステップ２４０３でハードウェア内で変数にデータが読み込まれるステート（ＳＴｊ）を調べる。

【００７３】このとき、対象となる変数が出力変数であった場合、ステップ２４０４でＣＰＵが変数に対応したデータを読み込むステート（ＳＴｉ）を調べる。次に、ステップ２４０５でハードウェア内で変数にデータが書き込まれるステート（ＳＴｊ）を調べる。ステップ２４０６でＳＴｊが等しいか等しくないかチェックする。ＳＴｉとＳＴｊが等しい場合、バッファは必要無いので、ステップ２４０７で、インターフェース回路詳細化情報テーブルのバッファ欄に“無”を登録する。

【００７４】ＳＴｉとＳＴｊが等しくない場合、データを保持しておく必要があるため、ステップ２４０８で、テーブルのバッファ欄に“有”を登録する。

【００７５】上記のような処理により、インターフェース回路詳細化情報テーブルが得られる。このテーブルよりインターフェース回路の詳細な情報が得られる。図２５に示した例では、インターフェース回路が２個の入力ポートと１個の出力ポートを持ち、それぞれのポートのビット幅は１６で、バッファは必要ないことがわかる。

【００７６】上記のようにして、１つのハードウェア／ソフトウェア分割候補のインターフェース回路の詳細化が得られる。

【００７７】（８）実施形態８

次に、本発明の第８の実施形態について説明する。この実施形態では第６の実施例で示した差分情報生成部１９０３で実行される差分情報生成を実現するものである。

【００７８】図２０に示した動作仕様の図２１に示したハードウェア／ソフトウェア分割と異なるハードウェア／ソフトウェア分割を図２６に示す。この分割では、動作仕様中ｆ（ａ，ｂ）とｇ（ｃ，ｄ，ｅ）がハードウェアで実現され、動作仕様の残りの部分はＣＰＵ上で実行されるソフトウェアで実現される。

【００７９】ハードウェアで実現されるｆ（ａ，ｂ）とｇ（ｃ，ｄ，ｅ）の動作を図２７に示す。ｆ（ａ，ｂ）の動作は第７の実施形態での動作と同じである。ｇ（ｃ，ｄ，ｅ）に対しては、これを実現するハードウェアは５クロック・サイクル掛かって動作を完了する。データの受渡しのタイミングについては、変数ｃに対応したデータがＣＰＵからステート１に受け渡され、ハードウェアが変数ｃにステート１でデータを読み込んでいる。変数ｄについても同様である。

【００８０】一方、変数ｅに関しては、変数ｅに対応したデータがＣＰＵからステート１で受け渡されるが、ハードウェアは変数ｅにステート２でデータを読み込んでいる。変数ｏｕｔの値がハードウェアからステート５に出力され、変数ｏｕｔに対応したデータをＣＰＵはステート５で読み込んでいる。

【００８１】差分情報生成部１９０３では、インターフェース回路詳細化部１９０２が生成したインターフェース回路詳細化情報に基づいて、ハードウェア／ソフトウェア分割に必要なインターフェース回路の差分情報を生成する。

【００８２】この差分情報生成のフローチャートを図２８に示す。初めに、ステップ２８０１で入力変数のリストＬＩを作成し、ステップ２８０２で入力ポートのリストＰＩを作成する。ステップ２８０３で変数を１つリストＬＩから取り出す。ステップ２８０４でポートを１つリストＰＩから取り出す。ステップ２８０５で変数にバッファが必要かどうか調べる。

【００８３】このとき必要な場合、ステップ２８０４で取り出したポートがバッファを持つかどうか調べる。持っていない場合、新たなポートをリストＰＩから取り出す。ポートがバッファを持っている場合、ステップ２８０７で対象となっている変数に対応したデータがＣＰＵから受け渡されるステートＳＴ１を調べる。

【００８４】ステップ２８０８でポートをステートＳＴｉで使用可能かどうか調べる。使用可能な場合、ステップ２８０９で使用状況テーブルのステートＳＴｉ欄に変数を登録する。

【００８５】使用状況テーブルの例を図２９に示す。使用不可能な場合、新たなポートをリストＰＩから取り出し、ステップ２８０４から新たに処理を始める。リスト

ＰＩから新たなポートが得られなかった場合、ステップ２８１０で差分情報テーブルの入力ポート行の個数欄の値を１だけインクリメントする。

【００８６】差分情報テーブルの例を図３０に示す。ステップ２８１１で変数にバッファが必要か調べ、バッファが必要な場合、ステップ２８１２で差分情報テーブルの入力ポート行のバッファ欄の値を１だけインクリメントする。ステップ２８１３で使用状況テーブルにポート行を１つ追加し、ステップ２８１４でその行の種類欄に入力を登録し、ステップ２８１５でステートＳＴｉ欄に変数を登録する。上記と同様な処理を出力変数に対しても行う。

【００８７】上記のようにハードウェア／ソフトウェア分割の差分情報が生成される。

【００８８】（９）実施形態９

次に、本発明の第９の実施形態について説明する。この実施形態では第６の実施例で示した評価部１９０４で実行される差分情報に基づいたハードウェア／ソフトウェア分割の評価を実現するものである。

【００８９】図２１に示したハードウェア／ソフトウェア分割で必要とされるインターフェース回路と、図２６に示したハードウェア／ソフトウェア分割で必要とされるインターフェース回路の差分情報を図３０に示す。この例では、図２６に示したハードウェア／ソフトウェア分割に必要なインターフェース回路が、入力ポートを１個、バッファを１個余分に必要であることを示している。

【００９０】図３０に示した差分情報からインターフェース回路の面積評価のフローチャートを図３１に示す。初めにステップ３１０１でインターフェース回路詳細化部１９０２が生成したインターフェース回路詳細化情報を読み込む。ステップ３１０２でこの情報に基づいて詳細化したインターフェース回路の面積Ａをインターフェース回路詳細化情報テーブルから得られる入力ポート数、出力ポート数、バッファ数とあらかじめ与えられているポート、バッファに面積に基づいて計算する。

【００９１】次に、ステップ３１０３で差分情報生成部１９０３が生成した差分情報を読み込む。ステップ３１０４で差分情報テーブルの入力ポート行の個数欄の値（ＮＩ）を読み込む。ステップ３１０５で差分情報テーブルの入力ポート行のバッファ欄の値（ＢＩ）を読み込む。ステップ３１０６で差分情報テーブルの出力ポート行の個数欄の値（ＮＯ）を読み込む。ステップ３１０７で差分情報テーブルの出力ポート行のバッファ欄の値（ＢＯ）を読み込む。

【００９２】これらの値は詳細化したインターフェース回路と現在評価対象となっているハードウェア／ソフトウェア分割に必要なインターフェース回路の入力ポート数、入力ポートのバッファ数、出力ポート数、出力ポートのバッファ数との差を表している。

【００９３】ステップ３１０８で上記４ステップで得られた入力ポート数の変化ＮＩ、出力ポート数の変化ＮＯ、バッファ数の変化ＢＩ＋ＢＯとあらかじめ与えられているポートとバッファの面積に基づいて、詳細化されたインターフェース回路と現在評価対象となっているハードウェア／ソフトウェア分割に必要なインターフェース回路との面積の差ｄＡを計算する。ステップ３１０９でステップ３１０２で計算した面積Ａと、ステップ３１０８で計算した面積の差ｄＡとから現在評価対象となっているハードウェア／ソフトウェア分割に必要なインターフェース回路の面積Ａ＋ｄＡを計算する。

【００９４】上記のようにハードウェア／ソフトウェア分割に必要なインターフェース回路の面積を、差分情報に基づいて見積もることにより、より正確な評価が短時間に可能となる。

【００９５】なお、上述した実施形態で述べた処理手順は、プログラムとして表現することができ、これをコンピュータ読み取り可能な記録媒体に記録することができる。この記録媒体とは、メモリ装置、磁気ディスク装置、光ディスク装置等、プログラムを記録することができるような装置が含まれる。

【００９６】

【発明の効果】本発明に係るシステムの設計支援装置及び設計支援プログラムを記録したコンピュータ読み取り可能な記録媒体によれば、処理機能をハードウェアとソフトウェアに分担して評価を行うことによってシステムの設計を行う際に、ハードウェアとソフトウェア間でデータの受け渡しを行うインターフェース手段による影響を考慮することによって、より正確なシステムの設計を容易且つ迅速に行うことができる。

【図面の簡単な説明】

【図１】本発明の第１の実施形態に係るハードウェア・コデザイン装置の構成図である。

【図２】第２の実施形態に係るハードウェア・コデザイン装置におけるスケジューリング・アーキテクチャ情報生成のフローチャートを示すものである。

【図３】　第２の実施形態に係るハードウェア・コデザイン装置における動作記述の例を示すものである。

【図４】第２の実施形態に係るハードウェア・コデザイン装置におけるハードウェア・ソフトウェア分割の例を示すものである。

【図５】第２の実施形態に係るハードウェア・コデザイン装置におけるスケジューリング結果の例を示すものである。

【図６】第２の実施形態に係るハードウェア・コデザイン装置におけるスケジューリング・アーキテクチャ情報の例１を示すものである。

【図７】第２の実施形態に係るハードウェア・コデザイン装置におけるスケジューリング・アーキテクチャ情報の例２を示すものである。

【図８】第２の実施形態に係るハードウェア・コデザイン装置におけるスケジューリング・アーキテクチャ情報の例３を示すものである。

【図９】第３の実施形態に係るハードウェア・コデザイン装置におけるコード系列情報生成のフローチャートを示すものである。

【図１０】第３の実施形態に係るハードウェア・コデザイン装置におけるコード系列の例を示すものである。

【図１１】第３の実施形態に係るハードウェア・コデザイン装置におけるコード系列情報の例１を示すものである。

【図１２】第３の実施形態に係るハードウェア・コデザイン装置におけるコード系列情報の例２を示すものである。

【図１３】第４の実施形態に係るハードウェア・コデザイン装置におけるモデル情報生成のフローチャートを示すものである。

【図１４】第４の実施形態に係るハードウェア・コデザイン装置におけるモデル情報の例１を示すものである。

【図１５】第４の実施形態に係るハードウェア・コデザイン装置におけるモデル情報の例２を示すものである。

【図１６】第５の実施形態に係るハードウェア・コデザイン装置における制約チェックのフローチャートを示すものである。

【図１７】第５の実施形態に係るハードウェア・コデザイン装置における制約評価値の計算例を示すものである。

【図１８】第５の実施形態に係るハードウェア・コデザイン装置における従来の制約評価値の計算例を示すものである。

【図１９】第６の実施形態に係るハードウェア・コデザイン装置におけるの構成図である。

【図２０】第７の実施形態に係るハードウェア・コデザイン装置における動作仕様の例を示すものである。

＊【図２１】第７の実施形態に係るハードウェア・コデザイン装置におけるハードウェア／ソフトウェア分割の例１を示すものである。

【図２２】第７の実施形態に係るハードウェア・コデザイン装置における図２１中のハードウェアの動作を示すものである。

【図２３】第７の実施形態に係るハードウェア・コデザイン装置におけるインターフェース回路詳細化フローチャートを示すものである。

【図２４】第７の実施形態に係るハードウェア・コデザイン装置におけるバッファの有無を判断するフローチャートを示すものである。

【図２５】第７の実施形態に係るハードウェア・コデザイン装置におけるインターフェース回路詳細化情報テーブルの例を示すものである。

【図２６】第８の実施形態に係るハードウェア・コデザイン装置におけるハードウェア／ソフトウェア分割の例２を示すものである。。

【図２７】第８の実施形態に係るハードウェア・コデザイン装置における図２６中のハードウェアの動作を示すものである。

【図２８】第８の実施形態に係るハードウェア・コデザイン装置における差分情報生成のフローチャートを示すものである。

【図２９】第８の実施形態に係るハードウェア・コデザイン装置における使用状況テーブルの例を示すものである。

【図３０】第９の実施形態に係るハードウェア・コデザイン装置における差分情報テーブルの例を示すものである。

【図３１】第９の実施形態に係るハードウェア・コデザイン装置における評価のフローチャートを示すものである。

【符号の説明】

【図３】

```
int f (int x)
{
  int result=0;
  for (int i=1; i<=10; i++)
    result=resutl+i*x;
  return result;
}

main ()
{
  int a,b;
  int x,y,z;

  a=...
  b=...
  x=f(a);
  y=f(b);
  ...
  z=x+y;
}
```

【図４】

ハードウェア	f (x)
ソフトウェア	main( )

【図１１】

ステップ数	n
関数呼び出し数	m

【図６】

アーキテクチャ	パイプライン
パイプライン・ピッチ	C
ステージ数	N
レイテンシー	L
データ投入サイクル	L/N

【図１２】

ステップ数	n
関数呼び出し数	2

【図１０】

```
...
add a,...
add b,...
call f(a)
call f(b)

...
ld x
ld y
...
```

【図１８】

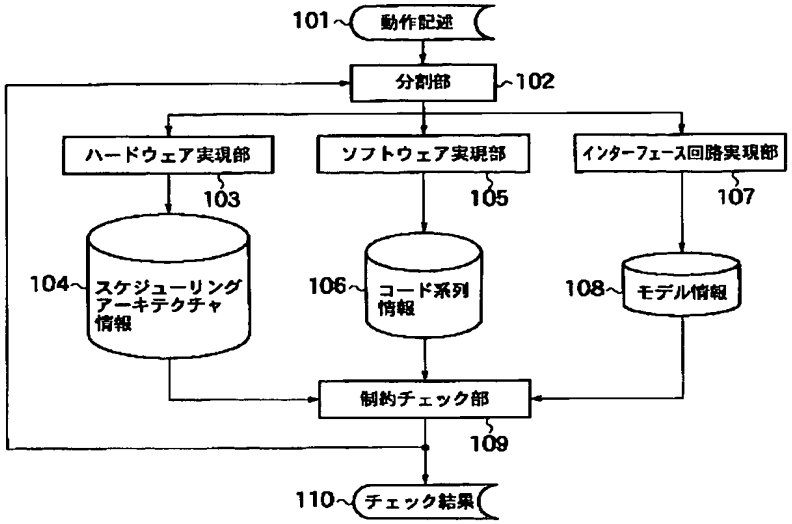
実行時間=$C_1 \times n$

【図１７】

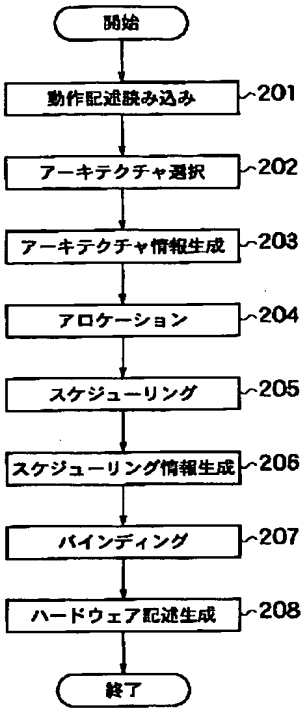
実行時間=$C_1 \times n+C_2 \times L$

$=C_1 \times n+C_2 \times 10$

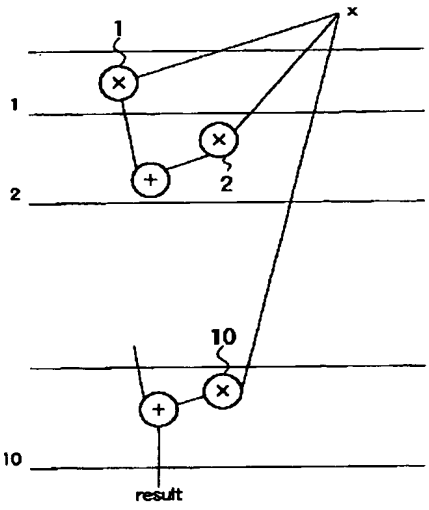
【図１】



【図２】



【図５】



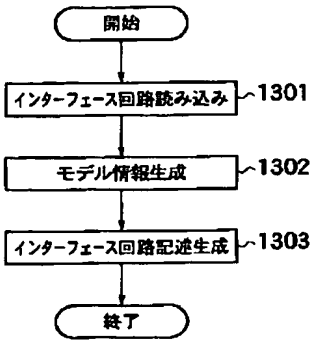
【図７】

アーキテクチャ	非パイプライン
パイプライン・ピッチ	C
ステージ数	1
レイテンシー	10
データ投入サイクル	10

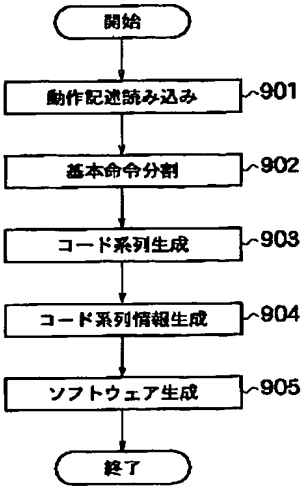
【図８】

アーキテクチャ	パイプライン
パイプライン・ピッチ	C
ステージ数	10
レイテンシー	10
データ投入サイクル	1

【図１３】



【図９】



【図１４】

プロトコル・タイプ	ハンドシェイク	キュー
結果が得られるまでの時間	L	L
データ投入サイクル	L	1

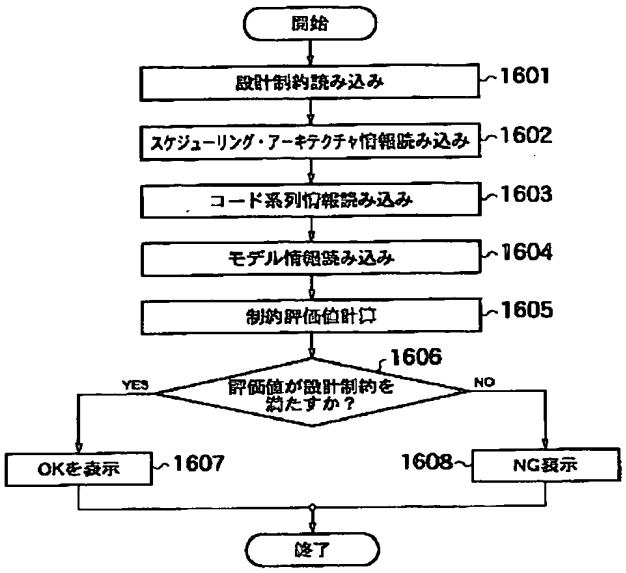
【図１５】

プロトコル・タイプ	ハンドシェイク	キュー
結果が得られるまでの時間	10	10
データ投入サイクル	10	1

【図２１】

ハードウェア	f(a,b)
ソフトウェア	動作仕様の残りの部分

【図１６】



【図２０】

```
main()
{
  int a,b,c,d,e;
  int x,y,z;
      ⋮
  x=f(a,b);
      ⋮
  y=g(c,d,e);
      ⋮
  z=x+y;
      ⋮
}
```

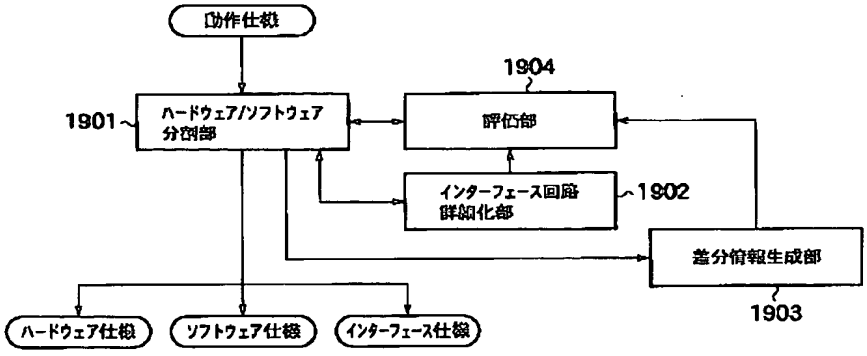
```
f(a,b)
{
  int out;
  out=a+b;
  return out;
}
```

```
g(c,d,e)
  int out;
  out=(c*d)-(d*e);
  return out;
}
```

【図２５】

ポート番号	種類	ビット幅	タイミング	バッファ	変数
1	入力	16	ステート1	無	a
2	入力	16	ステート1	無	b
3	出力	16	ステート3	無	out

【図１９】



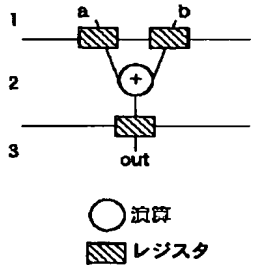
【図２６】

ハードウェア	f(a,b) g(c,d,e)
ソフトウェア	動作仕様の残りの部分

【図３０】

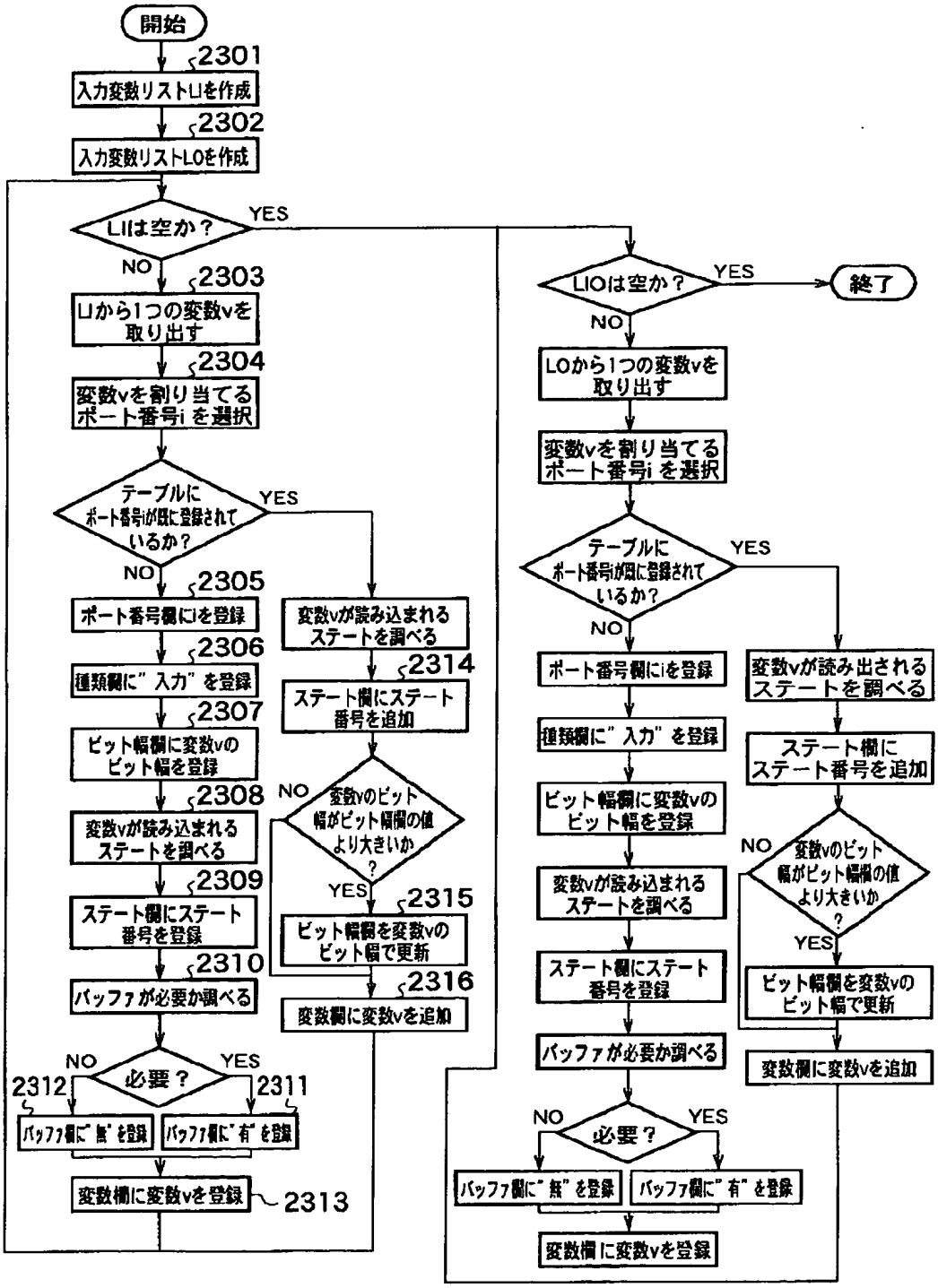
	個数	バッファ
入力ポート	+1	+1
出力ポート	±0	±0

【図２２】

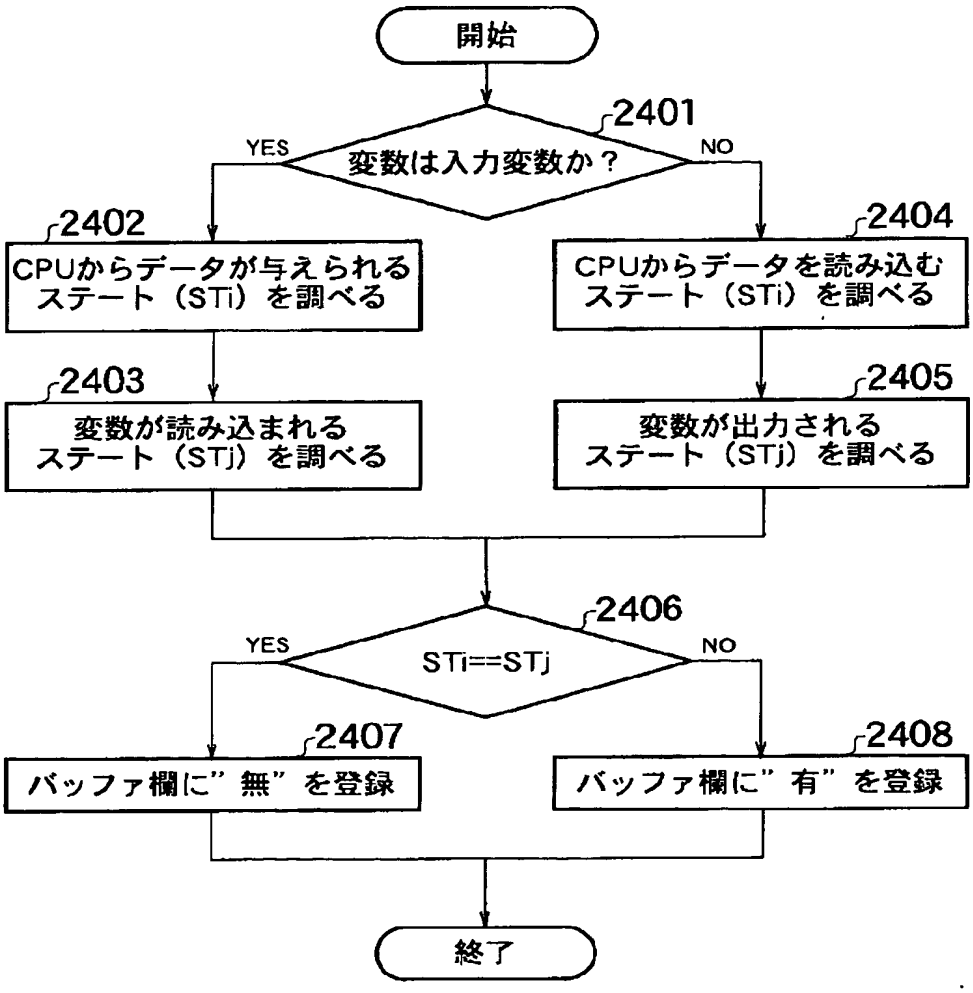


変数	CPUから（へ）データが出力（入力）されるタイミング
a	ステート1
b	ステート1
out	ステート3

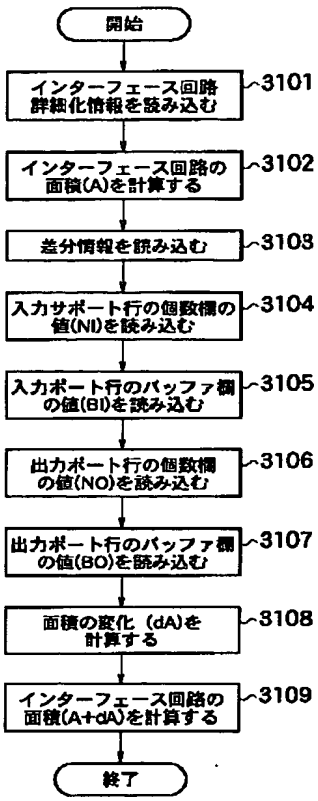
【図２３】


開始
2301
入力変数リストLIを作成
2302
入力変数リストLOを作成
LIは空か？
YES
NO
2303
LIから1つの変数vを取り出す
2304
変数vを割り当てるポート番号i を選択
テーブルにポート番号iが既に登録されているか？
YES
NO
2305
ポート番号欄にiを登録
2306
種類欄に" 入力" を登録
2307
ビット幅欄に変数vのビット幅を登録
2308
変数vが読み込まれるステートを調べる
2309
ステート欄にステート番号を登録
2310
バッファが必要か調べる
必要？
NO
YES
2312
2311
バッファ欄に" 無" を登録
バッファ欄に" 有" を登録
変数欄に変数vを登録
2313
変数vが読み込まれるステートを調べる
2314
ステート欄にステート番号を追加
変数vのビット幅がビット幅欄の値より大きいか？
NO
YES
2315
ビット幅欄を変数vのビット幅で更新
2316
変数欄に変数vを追加
LIOは空か？
YES
終了
NO
LOから1つの変数vを取り出す
変数vを割り当てるポート番号i を選択
テーブルにポート番号iが既に登録されているか？
YES
NO
ポート番号欄にiを登録
種類欄に" 入力" を登録
ビット幅欄に変数vのビット幅を登録
変数vが読み込まれるステートを調べる
ステート欄にステート番号を登録
バッファが必要か調べる
必要？
NO
YES
バッファ欄に" 無" を登録
バッファ欄に" 有" を登録
変数欄に変数vを登録
変数vが読み出されるステートを調べる
ステート欄にステート番号を追加
変数vのビット幅がビット幅欄の値より大きいか？
NO
YES
ビット幅欄を変数vのビット幅で更新
変数欄に変数vを追加

【図２４】



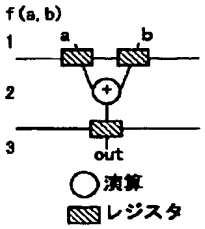
【図３１】



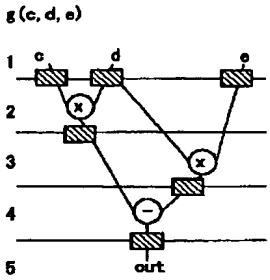
【図２９】

ポート番号	種類	ステート1	ステート2	ステート3	ステート4	ステート5
1	入力	c				
2	入力	d				
3	出力					out
4	入力	e				

【図２７】

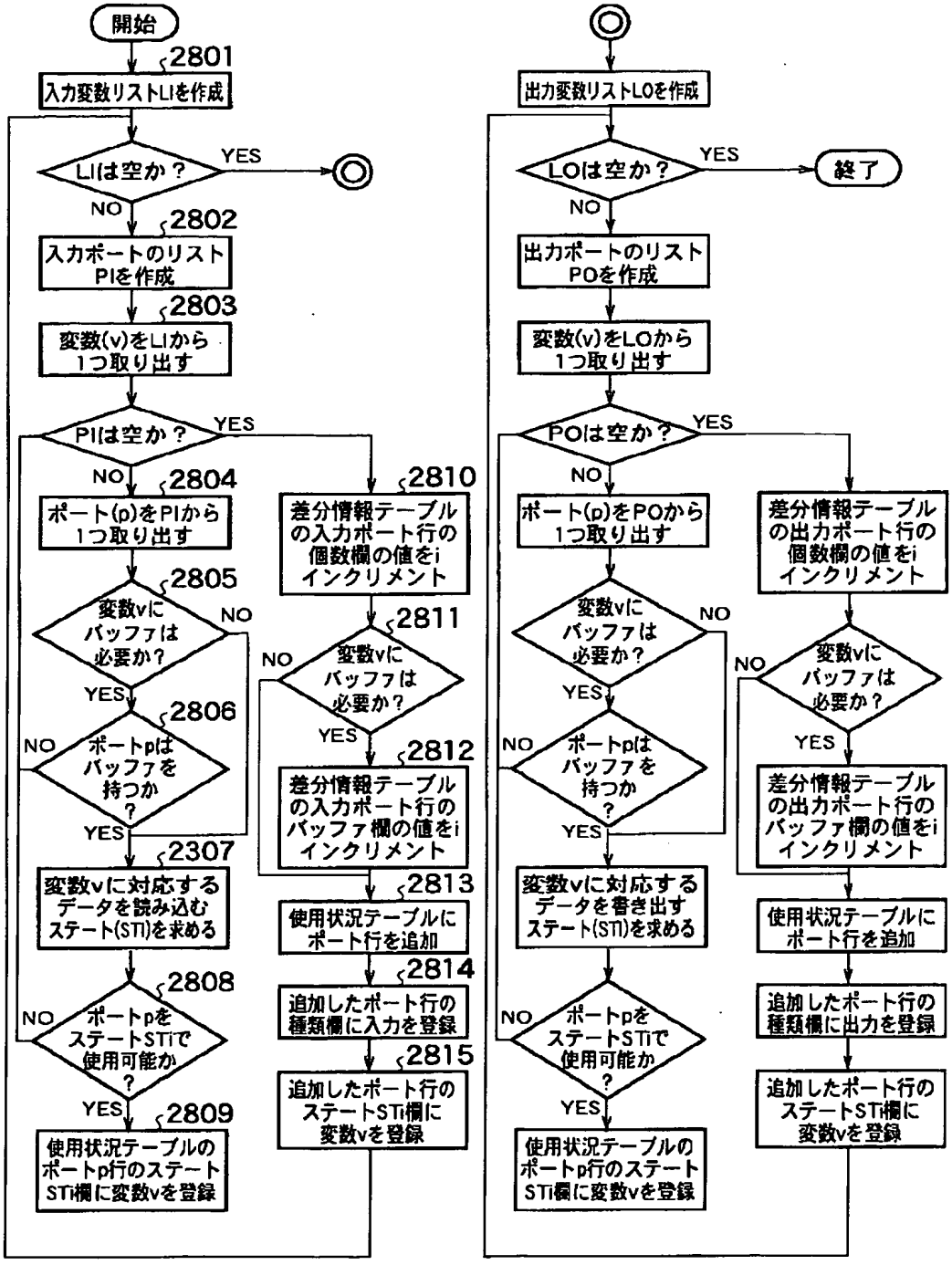


変数	CPUから（へ）データが出力（入力）されるタイミング
a	ステート1
b	ステート1
out	ステート3



変数	CPUから（へ）データが出力（入力）されるタイミング
c	ステート1
d	ステート1
e	ステート1
out	ステート4

【図２８】


開始
2801
入力変数リストLIを作成
LIは空か？
YES
NO
2802
入力ポートのリスト PIを作成
2803
変数(v)をLIから 1つ取り出す
PIは空か？
YES
NO
2804
ポート(p)をPIから 1つ取り出す
2805
変数vに バッファは 必要か？
NO
YES
2806
ポートpは バッファを 持つか ？
NO
YES
2307
変数vに対応する データを読み込む ステート(STI)を求める
2808
ポートpを ステートSTiで 使用可能か ？
NO
YES
2809
使用状況テーブルの ポートp行のステート STi欄に変数vを登録
2810
差分情報テーブル の入力ポート行の 個数欄の値をi インクリメント
2811
変数vに バッファは 必要か？
NO
YES
2812
差分情報テーブル の入力ポート行の バッファ欄の値をi インクリメント
2813
使用状況テーブルに ポート行を追加
2814
追加したポート行の 種類欄に入力を登録
2815
追加したポート行の ステートSTi欄に 変数vを登録
出力変数リストLOを作成
LOは空か？
YES
NO
終了
出力ポートのリスト POを作成
変数(v)をLOから 1つ取り出す
POは空か？
YES
NO
ポート(p)をPOから 1つ取り出す
変数vに バッファは 必要か？
NO
YES
ポートpは バッファを 持つか ？
NO
YES
変数vに対応する データを書き出す ステート(STI)を求める
ポートpを ステートSTiで 使用可能か ？
NO
YES
使用状況テーブルの ポートp行のステート STi欄に変数vを登録
差分情報テーブル の出力ポート行の 個数欄の値をi インクリメント
変数vに バッファは 必要か？
NO
YES
差分情報テーブル の出力ポート行の バッファ欄の値をi インクリメント
使用状況テーブルに ポート行を追加
追加したポート行の 種類欄に出力を登録
追加したポート行の ステートSTi欄に 変数vを登録